Randomized and Rank based Differential Evolution

Onay Urfalioglu and Orhan Arikan
Bilkent University
Department of Electrical and Electronics Engineering
06800 Ankara, Turkey
e-mail: {onay,oarikan}@ee.bilkent.edu.tr

Abstract

Many real world problems which can be assigned to the machine learning domain are inverse problems. The available data is often noisy and may contain outliers, which requires the application of global optimization. Evolutionary Algorithms (EA's) are one class of possible global optimization methods for solving such problems. Within population based EA's, Differential Evolution (DE) is a widely used and successful algorithm. However, due to its differential update nature, given a current population, the set of possible new populations is finite and a true subset of the cost function domain. Furthermore, the update formula of DE does not use any information about the fitnesses of the population. This paper presents a novel extension of DE called Randomized and Rank based Differential Evolution (R2DE) to improve robustness and global convergence speed on multimodal problems by introducing two multiplicative terms in the DE update formula. The first term is based on a random variate of a Cauchy distribution, which leads to a randomization. The second term is based on ranking of individuals, so that R2DE exploits additional information provided by the fitnesses. In experiments including non-linear dimension reduction by autoencoders, it is shown that R2DE improves robustness and speed of global convergence.

1 Introduction

Within the class of Evolutionary Algorithms (EA's), Differential Evolution (DE) [12, 16] is one the most robust, fast [17] and easily implementable methods. It has only three control parameters, including the population size. A striking property of DE is that it incorporates self adaptation by automatically scaling the search area on each phase of the global search process resulting in optimized efficiency. The main application domain of EA's is the optimization of multimodal functions. For many important problems such as those in the complexity class NP, the required number of function evaluations increases exponentially with the search space dimension. Therefore, the efficiency of an EA determines the practical limit at which applications based on those problems can be realized.

The proposed method called Randomized and Rank based Differential Evolution (R2DE) integrates two distinct concepts in producing a new population of solution candidates: randomization and the utilization of ranking. DE has the property that the set of possible proposal vectors, which contains all possible results of mutation and crossover given a population, is finite. Furthermore, the support of the distribution of the proposal vectors is finite too. The effect of the randomization is that these attributes become infinite. The second concept takes advantage of the fitness information of each individual. This information is not used in DE's mutation and crossover operators. We show experimentally that these concepts generally improve the efficiency of the global search when applied to DE.

In the literature, DE is subject of improvement in several publications. In two different works, Liu and Lampinen [9] and Brest, et al. [2], introduce methods for on-line self-adaptation of DE's control parameters for mutation and crossover. In [21], Teo applies self-adaptation to the population size. In [1], Ali and Törn propose auxiliary population and automatic calculation of the amplification coefficient. Tasoulis et al. [20] introduce parallel DE, where the population is divided into subpopulations, where each subpopulation is assign to a different processor node. In [15], Shi, et al., propose the so called cooperative coevolutionary differential evolution where multiple cooperating subpopulations are used and high dimensional search spaces are partitioned into smaller spaces.

Other methods improving DE are based on hybridization. In [19], Sun et al. propose a hybrid algorithm using an estimation of distribution method. This method is based on a probability model which is sampled from

978-0-7695-3926-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ICMLA.2009.29

to generate additional solution candidates. Noman and Iba [10] propose a local search to accelerate the fine tuning phase of DE based on fittest individual refinement which is a crossover-based local search. In [3], Fan and Lampinen introduce another local search - DE hybrid, which is called trigonometric mutation, in order to obtain a better tradeoff between convergence speed and robustness. Kaelo and Ali [8] introduce reinforcement learning based DE where different schemes for the generation of proposal vectors are proposed. Another interesting approach called Opposition Based Differential Evolution (ODE) based on oppositional numbers is presented by Rahnamayan et al. [14].

We compare the performance of the proposed approach to that of DE on scalable multimodal problems. We show the *tendency* of the global search efficiency of each method by increasing the number of dimensions of the search space or varying other complexity parameters, depending on the problem. Taking only one single dimension or complexity parameter into account is not enough and can lead to wrong conclusions, since some methods may be slower in a low dimensional setting but may become more efficient than the compared method in a higher dimension.

The paper is organized as follows. The following Section 2 briefly reviews DE. Section 3 introduces the proposed method R2DE. In Section 4, experimental results are shown and the paper is concluded in Section 5.

2 Brief Review of Differential Evolution

DE is one of the best general purpose evolutionary global optimization methods available. It is known as an efficient global optimization method for continuous cost functions. The optimization is based on a population of N_p solution candidates $\boldsymbol{x}_i, i \in \{1, ..., N_p\}$, also called individuals, where each individual has a position in the D-dimensional search space. Initially, the individuals are generated randomly according to a uniform distribution within the provided intervals of the search space. The population improves iteratively by generating a new position $\boldsymbol{u}$ for each individual $\boldsymbol{x}_{i,G}$ by

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{x}_{r_1,G} + F \cdot (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) && (1) \\ \boldsymbol{u} &= C\,(\boldsymbol{x}_{i,G},\, \boldsymbol{v}), && (2) \end{aligned}$$

where r_1, r_2, r_3 are pairwisely different random integers from the discrete set $\{1, ..., N_p\}$ and F is a weighting scalar. The vector $\boldsymbol{v}$ is used together with $\boldsymbol{x}_{i,G}$ in the crossover operation, denoted by $C()$. The crossover operator copies coordinates from both $\boldsymbol{x}_{i,G}$ and $\boldsymbol{v}$ in order to create the trial vector $\boldsymbol{u}$. C is provided with the probability C_r to copy coordinates from $\boldsymbol{x}_{i,G}$, whereby coordinates from $\boldsymbol{v}$ are copied with a probability of $1 - C_r$ to $\boldsymbol{u}$. Only if the new candidate $\boldsymbol{u}$ proves to have a lower cost it replaces $\boldsymbol{x}_{i,G}$, otherwise it is discarded.

DE includes an adaptive range scaling for the generation of solution candidates through the difference term in Equation (1). This leads to a global search with large step sizes in the case where the solution candidate vectors are widely spread within the search space due to a relatively large mean difference vector. In the case of a converging population, the mean difference vector becomes relatively small and this enables efficient fine tuning at the final phase of the optimization process. The crossover operator helps to increase the diversity of the population. In some problems, it can also speed up the convergence.

In case of regularly distributed local optima, the mutation scheme of DE in Eq. (1) is particularly advantageous due to its differential nature. During the convergence process, there is a high probability that individuals are located within the peaks of the local optima. Therefore, the difference vectors are generated approximately between the peaks of two selected local optima. In a mesh like distribution of the local optima, the resulting new position of an individual hits the area around the peak of another local optimum with high probability, depending on the weight factor F. Fig. 1 illustrates this property of DE's mutation scheme, in a one dimensional example. On the other

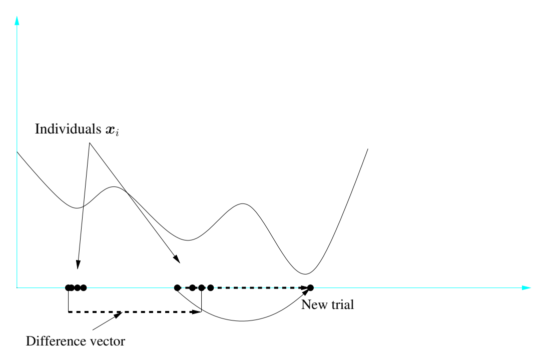

Figure 1. *In this 1-D example of **regularly distributed** local optima, the additive difference vectors yield, with high probability, new solution candidates which are located in the near vicinity of another local optimum.*

hand, this scheme can become inefficient on search spaces with non-regular structures where local optima have a non-regular distribution.

3 Randomized and Rank based Differential Evolution (R2DE)

The modifications of DE which make up R2DE are twofold. Two new multiplicative terms extend the update formula in Eqn. (1). The first term is a random variable λ which should be chosen to have heavy tails. Here, we will only consider the case where λ has Cauchy distribution, which has the following density:

$$f(\lambda) = \frac{1}{\pi(1+\lambda^2)},\ \lambda \in \mathbb{R}. \tag{3}$$

Its maximum is at zero, so that the majority of random variates from this distribution is concentrated at zero. Note that, due to its heavy tailed nature, the Cauchy distribution has no finite moments and it is much more likely to have samples which differ significantly from zero, in contrast to the normal distribution.

The second term α, which is in $(0,1]$ interval, is defined as

$$\alpha(\boldsymbol{x}_{r_1,G}) = 1 - \frac{k(\boldsymbol{x}_{r_1,G})}{N_p}, \tag{4}$$

where $k(\boldsymbol{x}_{r_1,G})$ is the rank of the individual $\boldsymbol{x}_{r_1,G}$. Assuming the global minimum is searched for, the best individual with minimal cost or fitness value has rank 0, whereas the worst individual has rank $N_p - 1$. This term reflects the fact that, on minimization of multimodal functions, the smaller the function values get, the more distant the regions with even lower function values become, in average. The update formula for the generation of trial vectors is given by

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{x}_{r_1,G} + F \cdot \lambda \cdot \alpha(\boldsymbol{x}_{r_1,G}) \cdot (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) && (5) \\ \boldsymbol{u} &= C\left(\boldsymbol{x}_{i,G},\ \boldsymbol{v}\right), && (6) \end{aligned}$$

where $\alpha(\boldsymbol{x}_{r_1,G})$ depends on $\boldsymbol{x}_{r_1,G}$ and λ is sampled independently for each individual $\boldsymbol{x}_{i,G}$ for each iteration.

4 Comparison Experiments

The experiments can be divided in two parts. The first part contains several scalable multimodal global optimization test problems which are common in the literature. The second part contains problems for non-linear dimension reduction using autoencoders [7]. In all experiments, unless mentioned otherwise, the utilized settings for the parameters are given by

- $F = 0.5$ (as in [1,2,9,14,18])
- $C_r = 0.9$ (as in [1,2,9,14,18])
- mutation strategy: DE/rand/1/bin (classic DE) (as in [2,13,14,18,19])
- value to reach (VTR) = 10^{-6}.

In the first part of the experiments, the following multimodal problems are used:

- Michalewicz function

$$\begin{aligned} f_1(\boldsymbol{x}) &= \sum_{j=1}^{D} -\sin(x_j) \cdot (\sin(j x_j^2/\pi))^{20} \\ x_j &\in [0,\pi],\ D \in \{5,...,12\}. \end{aligned}$$

D	VTR	D	VTR
5	-4.68765	6	-5.68765
7	-6.68088	8	-7.66375
9	-8.66014	10	-9.66014
11	-10.6574	12	-11.6495

- Perm function (D=4)

$$\begin{aligned} f_2(\boldsymbol{x}) &= \sum_{k=1}^{D}\left[\sum_{j=1}^{D}\left(j^k+\beta\right)\left(\left[\frac{x_j}{j}\right]^k - 1\right)\right]^2 \\ x_j &\in [-D,D],\ \beta \in \{4,5,...,13\}. \end{aligned}$$

- Perm0 function (D=4)

$$\begin{aligned} f_3(\boldsymbol{x}) &= \sum_{k=1}^{D}\left[\sum_{j=1}^{D}(j+\beta)\left(x_j^k - \left[\frac{1}{j}\right]^k\right)\right]^2 \\ x_j &\in [-1,1],\ \beta \in \{70,80,...,100\}. \end{aligned}$$

- Rastrigin function

$$\begin{aligned} f_4(\boldsymbol{x}) &= 10D + \sum_{j=1}^{D} x_j^2 - 10\cos(2\pi x_j) \\ x_j &\in [-5.12, 5.12],\ D \in \{9,...,16\}. \end{aligned}$$

- Schubert function

$$\begin{aligned} f_5(\boldsymbol{x}) &= \prod_{j=1}^{D}\sum_{k=1}^{5} k\cos((k+1)x_j + k) \\ x_j &\in [-10,10],\ D \in \{2,...,6\} \end{aligned}$$

D	2	3	4
VTR	-186.7309	-2709.1	-39303.6
D	5	6	
VTR	-570215.8	-8.2726·10^6	

- Schwefel function

$$\begin{aligned} f_6(\boldsymbol{x}) &= \sum_{j=1}^{D} -x_j \sin(\sqrt{|x_j|}) \\ x_j &\in [-500,500],\ D \in \{25,...,30\} \\ \mathrm{VTR}(D) &= -D \cdot 418.9829. \end{aligned}$$

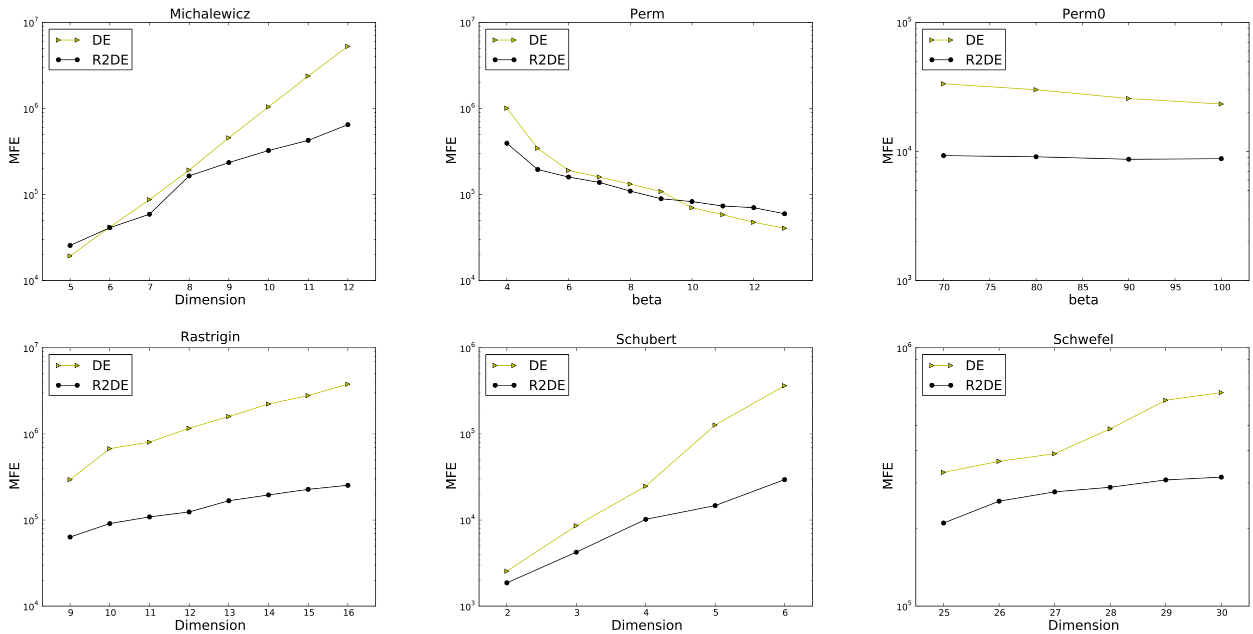

Figure 2. *Required mean function evaluations (MFEs) to find the global optimum with a robustness of $\rho = 0.99$. Note that smaller beta parameters in 'Perm' and 'Perm0' increase the complexity of the cost functions.*

For each problem, 100 independent optimization runs were carried out at different complexity settings such as space dimension or other function parameters. The task is to achieve a robustness of $\rho \approx 0.99$, i.e., at most one of the 100 runs may fail to find the global optimum. The global optimum is found when the VTR is reached. For each setting, the population size is adjusted individually to minimize the required mean function evaluations (MFE) and to meet the robustness constraint of $\rho = 0.99$. Fig. 2 shows the results of the comparisons between DE and R2DE. On each problem, R2DE outperforms DE regarding the required MFE. Moreover, the difference of the MFE's increases with the complexity settings. Table 1 shows the detailed measurements including the population sizes and the std.-deviations of the MFE's.

The second part of the experiments contains non-linear dimension reduction given two data sets. The first data set **sphere** contains 3-D points which are located on the hull of the unit-sphere, i.e., on a 2-D subset of the 3-D space. The data points $\boldsymbol{r}_j$ are generated by

$$\boldsymbol{r}_{5k+l} = \begin{pmatrix} \cos(k\pi/10)\sin(l\pi/10) \\ \sin(k\pi/10)\sin(l\pi/10) \\ \cos(l\pi/10) \end{pmatrix}, \; k, l = 0, ..., 4. \quad (7)$$

The utilized autoencoder is based on multi layer feedforward neural networks with sigmoidal neurons [4–6]. The structure of the networks is described by the 5-tuple

$$T = (n_0, n_1, n_2, n_1, n_0). \quad (8)$$

This means that the network has 5 layers. For the **sphere** data set, the input and output layers each have 3 neurons ($n_0 = 3$), the second and fourth layers have each n_1 and the third layer has $n_2 = 2$ neurons, yielding $T = (3, n_1, 2, n_1, 3)$. On each experiment, 25 inlier data points $\boldsymbol{r}_j$, $j = 1, ..., 25$ are used. The cost function to be minimized is

$$\Omega(\boldsymbol{\theta}) = \sum_{j=1}^{N} \sum_{k=1}^{K} (\boldsymbol{r}_{j,k} - f(\boldsymbol{\theta}, \boldsymbol{r}_j)_k)^2 , \quad (9)$$

where $f(\boldsymbol{\theta}, \boldsymbol{r}_j)$ represents the neural net mapping, $\boldsymbol{\theta}$ includes the parameters of the neural net and K is the dimension of the data points. In all remaining experiments following settings are used for both DE and R2DE:

- max. iterations = 10^4
- first experiment (no outliers): population size = 100 (MFE = 10^6), data set: $N = 25$ points
- second experiment (outliers): population size = 200 (MFE = $2{\cdot}10^6$), data set: $N = 25{+}10, 25{+}25, 25{+}40$ points
- $F = 0.5,\ C_r = 0.9$

Two experiments with 100 independent optimization runs each are carried out. In the first experiment, the Mean

Cost function	[Population size] MFE $\pm\ \sigma_{\text{MFE}}$ DE	R2DE	t-test P_t
Michalewicz [D=10]	[370] $1.04241{\cdot}10^6 \pm 116718$	[720] **324763** $\pm$ **17185**	$1.039{\cdot}10^{-82}$
Michalewicz [D=11]	[440] $2.37704{\cdot}10^6 \pm 292389$	[790] **426244** $\pm$ **22838**	$9.975{\cdot}10^{-85}$
Michalewicz [D=12]	[510] $5.27089{\cdot}10^6 \pm 630430$	[940] **648882** $\pm$ **34567**	$2.067{\cdot}10^{-88}$
Perm [β=6]	[450] 190814 $\pm$ 43699	[610] **159930** $\pm$ **32072**	$4.824{\cdot}10^{-8}$
Perm [β=5]	[800] 345192 $\pm$ 67983	[720] **195538** $\pm$ **40033**	$8.100{\cdot}10^{-43}$
Perm [β=4]	[2100] $1.00737{\cdot}10^6 \pm 229005$	[1400] **394501** $\pm$ **87772**	$5.779{\cdot}10^{-51}$
Perm0 [β=90]	[90] 25742 $\pm$ 4878	[30] **8714** $\pm$ **3822**	$1.273{\cdot}10^{-67}$
Perm0 [β=80]	[100] 30132 $\pm$ 6011	[30] **9109** $\pm$ **4659**	$7.429{\cdot}10^{-68}$
Perm0 [β=70]	[110] 33469 $\pm$ 7035	[30] **9300** $\pm$ **3974**	$1.400{\cdot}10^{-66}$
Rastrigin[D=14]	[200] $2.22585{\cdot}10^6 \pm 602941$	[350] **195531** $\pm$ **9376**	$4.917{\cdot}10^{-56}$
Rastrigin[D=15]	[220] $2.79051{\cdot}10^6 \pm 524350$	[380] **227305** $\pm$ **11667**	$3.612{\cdot}10^{-71}$
Rastrigin[D=16]	[240] $3.78711{\cdot}10^6 \pm 825896$	[400] **253272** $\pm$ **12782**	$1.120{\cdot}10^{-65}$
Schubert [D=4]	[40] 24724 $\pm$ 5307	[40] **10188** $\pm$ **1836**	$1.807{\cdot}10^{-51}$
Schubert [D=5]	[80] 126971 $\pm$ 23519	[50] **14717** $\pm$ **2584**	$4.827{\cdot}10^{-71}$
Schubert [D=6]	[120] 363317 $\pm$ 76073	[70] **29494** $\pm$ **5866**	$4.234{\cdot}10^{-67}$
Schwefel [D=28]	[170] 485841 $\pm$ 69941	[360] **288518** $\pm$ **14304**	$1.330{\cdot}10^{-50}$
Schwefel [D=29]	[190] 626901 $\pm$ 113176	[370] **308051** $\pm$ **13207**	$1.415{\cdot}10^{-49}$
Schwefel [D=30]	[190] 671090 $\pm$ 121122	[370] **315795** $\pm$ **12798**	$4.555{\cdot}10^{-51}$

Table 1. *Comparison table at robustness $\rho = 0.99$. Better results are shown in boldface. The t-test column contains the probability P_t that the difference of the MFE-means is due to chance. All t-tests clearly reject the hypotheses $MFE_{R2DE} = MFE_{DE}$ and $MFE_{R2DE} > MFE_{DE}$. Note that in the second and third columns the bracketed terms correspond to population size.*

Squared Errors (MSE's) are determined for different layer sizes (n_1) of the autoencoder network. In the second experiment, the utilized network has the structure $T = (3, 6, 2, 6, 3)$, yielding 74 degrees of freedom. The **sphere** data set is modified by adding zero mean Gaussian noise with std.-deviation $\sigma = 0.001$ and outlier data points. The outliers are sampled from a uniform distribution, each coordinate within $[-4, 4]$. For different outlier rates β, defined by

$$\beta = \frac{\text{outlier count}}{\text{total number of points}}, \tag{10}$$

the MSE's of the *inliers* are determined. In case of outliers, the optimization is based on the following robust cost function [11,22,23]

$$\Omega_R(\boldsymbol{\theta}) = \sum_{j=1}^{N}\sum_{k=1}^{K} -\log\left\{\frac{0.5e^{\frac{\left(\boldsymbol{r}_{j,k}-f(\boldsymbol{\theta},\boldsymbol{r}_j)_k\right)^2}{2\cdot 10\cdot\sigma^2}}}{\sqrt{2\pi\cdot 10\cdot\sigma^2}} + \frac{0.5}{8}\right\}. \tag{11}$$

The results of both experiments are shown in Fig. 3. As the size of the neural network is increased, the MSE produced by DE also increases clearly, while the proposed R2DE method yields the same small MSE for all three settings. The introduction of outliers and the utilization of the robust const function leads to increased MSE's on both DE and R2DE. However, R2DE clearly outperforms DE also in this case on all three outlier rates.

5 Conclusions

A novel Evolutionary Algorithm, Randomized and Rank based Differential Evolution (R2DE), is presented as a modification to the well known Differential Evolution (DE) method. According to the presented experimental results, R2DE outperforms DE in common global optimization problems regarding the required mean function evaluations (MFE's), where R2DE requires less MFE than DE. Futhermore, the MFE-differences increase with the complexity of the problem.

Experiments based on non-linear dimension reduction problems using autoencoder networks show that R2DE is capable of achieving better results regarding the Mean Squared Error (MSE).

6 Acknowledgement

This work was funded by the Turkish Scientific and Technical Research Council (TUBITAK).

References

[1] M. M. Ali and A. Törn. Population set-based global optimization algorithms: some modifications and numerical studies. *Comput. Oper. Res.*, 31(10):1703–1725, 2004.

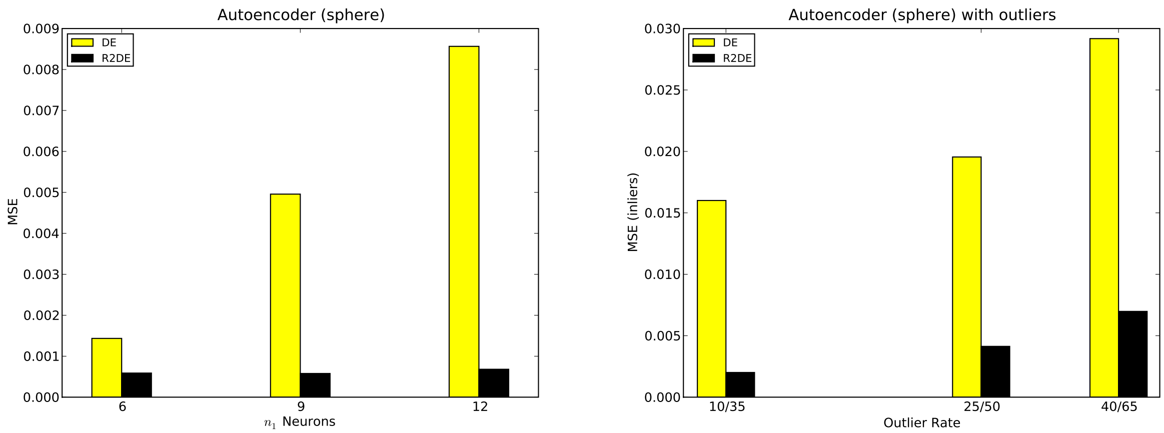

Figure 3. *Optimization results for the autoencoder problem on the* **sphere** *data set. Mean Squared Error (MSE) over the n_1 neuron size (see (8)) (left), and MSE over outlier rate β (see (10)) (right).*

[2] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *Evolutionary Computation, IEEE Transactions on*, 10(6):646–657, Dec. 2006.

[3] H.-Y. Fan and J. Lampinen. A trigonometric mutation operation to differential evolution. *J. of Global Optimization*, 27(1):105–129, 2003.

[4] K. Gurney. *An Introduction to Neural Networks*. Ucl Pr Ltd, 1997.

[5] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2nd edition, July 1998.

[6] J. A. Hertz, A. S. Krogh, and R. G. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley, Redwood City, CA, 1991.

[7] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[8] P. Kaelo and M. M. Ali. Probabilistic adaptation of point generation schemes in some global optimization algorithms. *J. Optim. Methods Softw.*, 21(3):343–357, 2006.

[9] J. Liu and J. Lampinen. A fuzzy adaptive differential evolution algorithm. *Soft Comput.*, 9(6):448–462, 2005.

[10] N. Noman and H. Iba. Enhancing differential evolution performance with local search for high dimensional function optimization. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 967–974, New York, NY, USA, 2005. ACM.

[11] O. Urfalioglu, P. Mikulastik, and I. Stegmann. Scale invariant robust registration of 3d-point data and a triangle mesh by global optimization. In *Proceedings of Advanced Concepts for Intelligent Vision Systems*, 2006.

[12] K. V. Price. Differential evolution: a fast and simple numerical optimizer. In *Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS*, pages 524–527. IEEE Press, New York. ISBN: 0-7803-3225-3, June 1996.

[13] K. V. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag, Berlin, Germany, 2005.

[14] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama. Opposition-based differential evolution. *IEEE Trans. Evolutionary Computation*, 12(1):64–79, 2008.

[15] Y.-J. Shi, H.-F. Teng, and Z.-Q. Li. Cooperative co-evolutionary differential evolution for function optimization. In *Proc. 1st Int. Conf. Advances in Natural Comput.*, pages 1080–1088, Changsha, China, Aug. 2005.

[16] R. Storn and K. Price. Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. *Technical Report TR-95-012, ICSI*, Mar. 1995.

[17] R. Storn and K. Price. Minimizing the real functions of the icec'96 contest by differential evolution. In *IEEE International Conference on Evolutionary Computation*, pages 842–844, Nagoya, May 1996.

[18] R. Storn and K. Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, 1997.

[19] J. Sun, Q. Zhang, and E. P. Tsang. De/eda: A new evolutionary algorithm for global optimization. *Information Sciences*, 169(3-4):249–262, 2005.

[20] D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis. Parallel differential evolution.

[21] J. Teo. Exploring dynamic self-adaptive populations in differential evolution. *Soft Comput.*, 10(8):673–686, 2006.

[22] O. Urfalioglu. Robust estimation of camera rotation, translation and focal length at high outlier rates. In *Proc. International Canadian Conference on Computer and Robot Vision*, pages 464–471, May 2004.

[23] A. Weissenfeld, O. Urfalioglu, K. Liu, and J. Ostermann. Robust rigid head motion estimation based on differential evolution. In *Proceedings of International Conference on Multimedia and Expo*, 2006.